```
//Angry Robots
//ITCS and Physics - Integrated Projectile Project

import java.text.DecimalFormat;
import edu.fcps.karel2.Display;
import edu.fcps.karel2.Robot;
import javax.swing.JOptionPane;

public class AngryRobots2Extension {

   // global variables
   // These can be used in any method in this program without having to pass the variable as a
parameter
   // Note that Ay and Ax are declared final, they cannot be changed
   public static final double Ay = -9.8;
   public static final double Ax = 0;
   public static double v0 = 0;
   public static double angleDegrees = 0;
   public static double y = 0.0;
   public static double x = 0.0;
   public static double v0x = 0.0;
   public static double v0y = 0.0;
   public static double t = 0.0;
   public static double dt = 0.1;
   public static double dY = 0.0;
   public static double airResistance = 0.0;


   public static void optimalAngle(){

    x = 0.0;
    y = 0.0;
    t = 0.0;

   v0x = v0 * Math.cos(Math.toRadians(angleDegrees));
   v0y = v0 * Math.sin(Math.toRadians(angleDegrees));

   while(y >= 0.0)
   {

      x = v0x * t;
      y = v0y * t + (Ay * (Math.pow(t,2))) * 1/2;
      t += dt;
```

```
  }

 }
  public static void plotPoints(double maxAngle)
  {
     for(double tempAngle = (maxAngle - 30.0); tempAngle <= (maxAngle + 30.0); tempAngle
+=15.0)
     {
        x = 0.0;
        y = 0.0;
        t = 0.0;

        v0x = v0 * Math.cos(Math.toRadians(tempAngle));
        v0y = v0 * Math.sin(Math.toRadians(tempAngle));



       while(y >= 0.0)
       {
          if(airResistance > 0)
          {
            x = (v0x * t) / airResistance;
            y = ((v0y * t + (Ay * (Math.pow(t,2))) * 1/2) + dY) / airResistance;
          }
          else
          {
             x = (v0x * t);
             y = v0y * t + (Ay * (Math.pow(t,2))) * 1/2 + dY;
          }

          t += dt;


         int xPlot = (int) x;
         int yPlot = (int) y;

         Robot john = new Robot(xPlot, yPlot, Display.NORTH, 0);
    }
  }
 }


  public static void main(String[] args){
    // Open default map and set speed
```

```
    Display.setSize(100, 100);
    Display.setSpeed(0);

    DecimalFormat commaFormat;
    commaFormat = new DecimalFormat("#.###");



    // User input to define initial velocity and launch angle
    // the parseDouble method converts the string input to a double
    v0 = Double.parseDouble(JOptionPane.showInputDialog("Please enter an initial velocity in
m/s"));
    dY = Double.parseDouble(JOptionPane.showInputDialog("Please enter a starting height
above the ground in Meters"));
    airResistance = Double.parseDouble(JOptionPane.showInputDialog("Please enter the
drag/air resistance for the projectile in Newtons"));



    // TODO: Calculate and plot x/y positions of the projectile as long as it remains above
ground



      double maxRange = 0.0;
      double maxAngle = 0.0;

      for(angleDegrees = 0; angleDegrees<=90; angleDegrees++)
      {
        optimalAngle();

        if(airResistance > 0)
        {
          System.out.println("For launch angle: " +
commaFormat.format(angleDegrees/airResistance) + " , Range is: " +
commaFormat.format(x/airResistance));
        }

        else
        {
          System.out.println("For launch angle: " + commaFormat.format(angleDegrees) + " ,
Range is: " + commaFormat.format(x));

        }
```

```
        if(x > maxRange && airResistance == 0)
        {
          maxRange = x;
          maxAngle = angleDegrees;
        }

        else if(x > maxRange && airResistance > 0)
        {
          maxRange = x / airResistance;
          maxAngle = angleDegrees / airResistance;
        }

      }

      System.out.println("Optimal angle for maximum range is: " +
commaFormat.format(maxAngle));
      System.out.println("For an initial velocity of " + commaFormat.format(v0) + ", max range
was "  + commaFormat.format(maxRange));
      System.out.println("Starting Height is: " + dY);
      System.out.println("Air Resistance on Particle is: " + airResistance);


      plotPoints(maxAngle);




    }

  }
```